In Proceedings of the 6th International Symposium on Networks-on-Chip (NOCS 2012)⋆

CCNoC: Specializing On-Chip Interconnects for Energy Efficiency in Cache-Coherent Servers

Stavros Volos[§], Ciprian Seiculescu[†], Boris Grot[§], Naser Khosro Pour[‡], Babak Falsafi[§], and Giovanni De Micheli[†]

[§]EcoCloud [†]LSI [‡]ELAB
École Polytechnique Fédérale de Lausanne

***Abstract*—Manycore chips are emerging as the architecture of choice to provide power efficiency and improve performance, while riding Moore's Law. In these architectures, on-chip interconnects play a pivotal role in ensuring power and performance scalability. As supply voltages begin to level off in future technologies, chip designs in general and interconnects in particular will require specialization to meet power and performance objectives.**

In this work, we make the observation that cache-coherent manycore server chips exhibit a duality in on-chip network traffic. Request traffic largely consists of simple control messages, while response traffic often carries cache-block-sized payloads. We present Cache-Coherence Network-on-Chip (CCNoC), a design that specializes the NoC to fit the demands of server workloads via a pair of asymmetric networks tuned to the type of traffic traversing them. The networks differ in their datapath width, router microarchitecture, flow control strategy, and delay. The resulting heterogeneous CCNoC architecture enables significant gains in power efficiency over conventional NoC designs at similar performance levels. Our evaluation reveals that a 4x4 mesh-based chip multiprocessor with the proposed CCNoC organization running commercial server workloads is 15-28% more energy efficient than various state-of-the-art single- and dual-network organizations.

I. Introduction

Today's server chips feature up to one hundred cores [20], and as chip integration levels keep increasing, future chips are expected to accommodate hundreds of cores [11]. Manycore chips rely on a network-on-chip (NoC) to lower design complexity and improve scalability. Recent research has identified high NoC power consumption as a significant obstacle in a quest for efficient manycore chips [5]. For example, in the MIT RAW processor, NoC power accounts for 40% of the overall chip power [22]. With supply voltages leveling off [7], the key design criteria for chips in general and NoCs in particular will be power.

A number of techniques have been proposed for improving NoC power consumption [1, 4, 13, 15, 23]. While the majority of these approaches focus on improved efficiency in the context of a single on-die network, Balfour and Dally [1] advocate using multiple networks as a practical way to improve performance, power, and area in tiled chip multiprocessors (CMPs). Using simple read/write messaging protocols, Balfour and Dally conclude that a homogeneous dual-network NoC, in which the two networks use identical microarchitectures and datapath width, balances the load among short and long messages and is preferred to a heterogeneous design.

Whereas prior work examined NoC efficiency in general-purpose chips, we target efficiency through network-level specialization in the context of server CMPs. Server chips rely on cache-coherent architectures for software transparency, and for facilitating software development and porting. Our analysis of commercial server workloads in a cache-coherent CMP reveals that network traffic does not follow simple read/write messaging protocols, commonly assumed in prior work, including Balfour and Dally [1]. In fact, traffic is highly skewed among short and long messages with (short) request messages primarily consisting of block fetch requests and clean replacement notifications and (long) response messages carrying a cache block.

These observations motivate us to propose *Cache-Coherence Network-on-Chip (CCNoC)*, a heterogeneous dual-network architecture for manycore server chips. CCNoC optimizes power and performance based on the characteristics of the two dominant message classes. The networks are asymmetric in their datapath width and router architecture. The *request* network is optimized for short messages, and thus it has a narrow datapath. As requests (e.g., reads) and the associated coherent requests (e.g., downgrades) travel through the same network, the network relies on virtual channels to segregate these message classes for deadlock avoidance. In contrast, the *response* network does not require any virtual channels and is customized for cache block transfers via a wide datapath and low-complexity wormhole routers.

We use *Flexus* [24] for cycle-accurate full-system multiprocessor simulation running commercial server workloads and augment it with custom power models to show that:

- Network traffic in server workloads is highly skewed among short and long messages. Short requests account for 94% of all requests and long responses account for 95% of all responses;
- Unlike NoCs for CMPs with simple messaging protocols that favor homogeneous networks, NoCs for cache-coherent CMPs require specialization and heterogeneity to best take advantage of the network traffic.
- CCNoC is more energy efficient than various state-of-the-art single- and dual-network organizations by 15-28%.

II. Background

We first describe the chip architecture that we consider in this paper. Next, we qualitatively explain how traffic duality

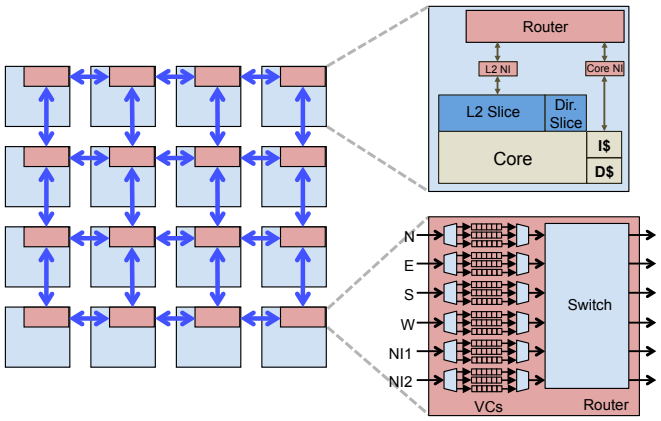

Figure 1: Tiled processor with mesh topology. Each network router uses three virtual channels (VCs); Request, Coherence request, and Response. The network router has six ports: N(orth), S(outh), W(est), E(ast), NI1 for the L1 controllers, and NI2 for the L2 slice.

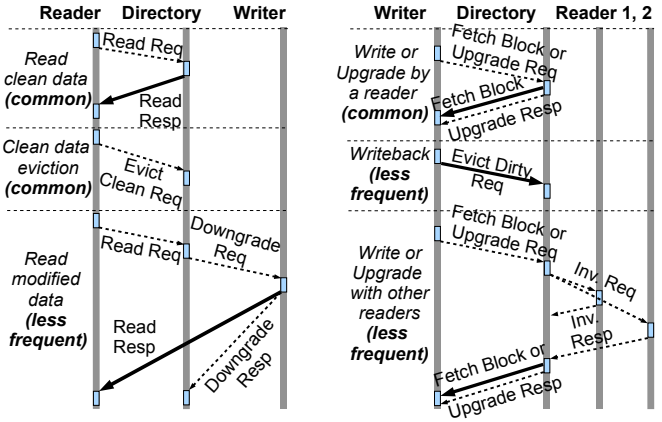

Figure 2: Read (left) and Write (right) protocol activity.

arises in cache-coherent CMPs and, finally, describe protocol-level deadlock issues.

Figure 1 depicts a canonical cache-coherent tiled CMP chip architecture. Each tile consists of a processor core with L1 data and instruction caches, an L2 slice, a directory slice, a network router, and two network interfaces (NIs). We assume a shared L2 cache organization, in which each L2 slice is a part of a shared last-level cache (LLC) and cache blocks are address-interleaved among the L2 slices. While the shared organization is preferred for server workloads, as it is more effective at capturing their large instruction and data footprints, our results equally hold for systems with a private LLC.

An invalidation-based directory protocol maintains coherence among the L1 caches. We assume, without loss of generality, a duplicate-tag directory scheme where each directory slice is responsible for the same range of address-interleaved cache blocks as the co-located L2 slice. The choice of directory encoding [26] may also affect the overall network traffic, but does not fundamentally affect the breakdown of the request and response message types.

Each tile uses two NIs, one for the core (i.e., L1 controllers) and the other for the L2 slice and the directory controller. This allows parallel accesses to the tile's caches and directory from remote cores. We assume a router architecture with four network ports and two local ports (connected to the NIs).

A. *Coherence protocol activity*

Coherence protocols were introduced as a way to ensure that any request for a cache block will get the most recent state of that cache block. Figure 2 depicts the protocol transitions for reading (data and instruction) and writing into cache blocks.

Figure 2 (left) shows the communication between a reader core, a directory slice, and a potential writer. Control messages are short and depicted with narrow dashed lines. Data messages carry a cache-block sized payload and are depicted with thick solid lines. In the common case, a reader sends a request for the read-only copy of a cache block, followed by a response from the L2 cache with the data. Similarly, to keep the directory up-to-date with sharer information, clean cache block replacements are also notified with small request messages. In the less frequent case of a read from an active writer of a block, the protocol implements a 3-hop transition. The read request is forwarded to the writer, which then responds directly to the reader and the directory with a data message and a notification response, respectively. Thus, most requests (reads or eviction notifications) for clean blocks are short messages and most responses carry a cache block.

Figure 2 (right) depicts the protocol transitions for writing into cache blocks. In general, write requests (i.e., fetch-block or upgrade requests) are less frequent. Moreover, in server workloads, writebacks account for a negligible fraction of the overall traffic because data are rarely updated and instructions are virtually never modified at runtime [6]. Finally, unlike cache block fetches, writebacks are not latency sensitive and therefore do not impact performance directly.

Even among the writes, there are common transitions that fit the duality in traffic. For example, write misses to blocks that are not actively shared have the same request/response behavior as reads of clean blocks. Other transitions include upgrade requests to a non-shared block, requiring a short request message and a short response message as well. Among write requests, those involving other readers and consequently a large number of control messages for both requests and responses are quite rare. In server workloads, data sharing and migration across threads happen over large windows of time – well beyond a typical L1 residency period [6]. As a result, writers rarely modify blocks shared by other cores [10].

Commercial server [6] and emerging scale-out cloud [3] applications are optimized for high reuse in the L1 data cache, while their instruction working sets exceed the L1 instruction cache capacity. Therefore, in server CMPs, coherence activity is dominated by short requests (clean data and instruction block fetches) and the associated long responses.

B. *Protocol-level deadlock avoidance*

Various message classes co-exist in cache-coherence protocols (i.e., requests, coherence requests, and responses). Consequently, protocol-level cyclic dependencies and deadlocks

(a)

CMP Size	16-core for server workloads 8-core for multiprogrammed workloads
Processing Cores	UltraSPARC III ISA; 2GHz 8-stage pipeline 2-wide dispatch / retirement, OoO
L1 Caches	Split I/D, 32KB 4-/2-way, 1-/2-cycle load-to-use 3 ports, 32 MSHRs, 8-entry victim cache
L2 NUCA Cache	512KB per core, 8-way, 10-cycle latency, 64-byte lines, 1 port, 32 MSHRs, 16-entry victim cache
Main Memory	3 GB total memory, 45 ns access latency
Memory Controller	one per 4 cores, round-robin page interleaving

(b)

Online Transaction Processing (TPC-C)	
DB2	100 warehouses (10 GB), 64 clients, 450 MB buffer pool
Oracle	100 warehouses (10 GB), 16 clients, 1.4 GB SGA
Decision Support Systems (TPC-H)	
DB2	Mixed, Queries: 1, 6, 13, 16; 1 GB buffer pool
Web Server (SPECweb99)	
APACHE	16K connections, fastCGI, worker threading model
ZEUS	16K connections, fastCGI
Multiprogrammed (SPEC CPU2000)	
SPEC2K	2 copies from each of gcc, twolf, mcf, art; reference input

Table I: System parameters for the 16-core and 8-core CMPs (a) and application parameters (b).

may occur due to messages sharing network resources. To avoid protocol-level deadlocks, conventional NoCs partition the physical resources at each router's input port among multiple virtual channels to allow independent routing of different message types. An alternative organization consists of multiple physical networks, with a dedicated network for each message class. In both cases, deadlock is avoided by routing messages of each class on a dedicated network (virtual or physical), thereby preventing the formation of cyclic dependencies across message classes.

III. Characterization of Network Traffic

A. Methodology

We use *Flexus* [24] for cycle-accurate full-system simulation of a tiled CMP executing unmodified applications and operating systems. *Flexus* extends the *Virtutech Simics* functional simulator with timing models of processing tiles with out-of-order cores, a detailed cache hierarchy, memory controllers, and a NoC. We simulate a tiled CMP similar to the one described in Figure 1 with a shared LLC.

Table I (a) summarizes our system architecture. We model a server chip composed of 16 cores and an 8 MB last-level cache. Recent work has demonstrated that cache capacities up to 4-8 MB are beneficial for capturing the instruction footprint and the small amount of shared data in server workloads [6, 8]. Cache capacities beyond 8 MB have a much lower utility due to the enormous memory footprints of these applications. We model a distributed (NUCA) LLC with 512 KB at each tile, with cache coherence based on the MESI protocol.

Our simulated system runs the *Solaris 8* operating system and executes the workloads listed in Table I (b). We include a wide range of commercial server workloads from the domains of online transaction processing, decision support systems, and web servers. We use the TPC-C v3.0 OLTP benchmark [21] on IBM DB2 v8 ESE and Oracle 10g Enterprise Database Server. We run a mix of queries 1, 6, 13, and 16 from the TPC-H benchmark [21] on DB2. Queries 1 and 6 are scan-bound, Query 16 is join-bound, and Query 13 exhibits a hybrid behavior. To evaluate web server performance, we use the SPECweb99 benchmark on Apache HTTP Server v2.0 and Zeus Web Server v4.3. We use a separate client system to drive the web servers, and hence do not include client activity in our measurements.

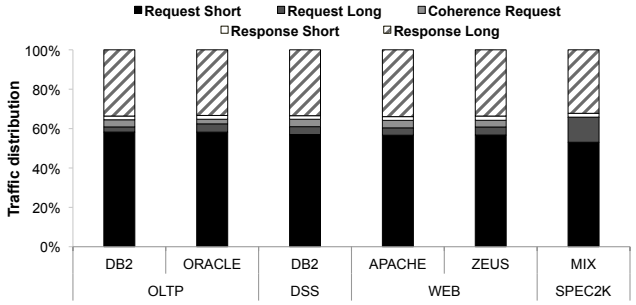

Figure 3: Network traffic distribution in server and multiprogrammed workloads.

For comparison, we also simulate a multiprogrammed workload that consists of SPEC CPU2000 applications running the reference input set. Because desktop applications lack concurrency and do not benefit from many-core execution substrates [2], we model an 8-core CMP for this study.

B. Characterization of network traffic

Figure 3 illustrates the distribution of short and long messages across on-chip request, coherence request, and response messages. The request traffic primarily consists of instruction fetches, cache-block reads, and clean eviction requests (i.e., short messages). Across server workloads, short messages account for 94% of the request traffic. In OLTP and Web workloads, which have big instruction footprints, instruction block requests dominate the request traffic. In DSS and the multiprogrammed workload, all with small instruction footprints, read requests are the majority of the request traffic.

In server workloads, the writeback traffic accounts only for 6% of the request traffic, implying that the clean eviction requests dominate the dirty eviction requests. In contrast, in the multiprogrammed workload, the writeback traffic accounts for 19% of the request traffic.

Coherence requests (i.e., invalidate and downgrade requests) are short and account for a small fraction of the request traffic. For the server workloads, coherence requests account for only 5% of the request traffic. The multiprogrammed workload does not exhibit such traffic because each core runs a different application and hence there is not any sharing among the cores.

Figure 3 indicates that the response traffic is also highly skewed. On average, long response messages account for 95%

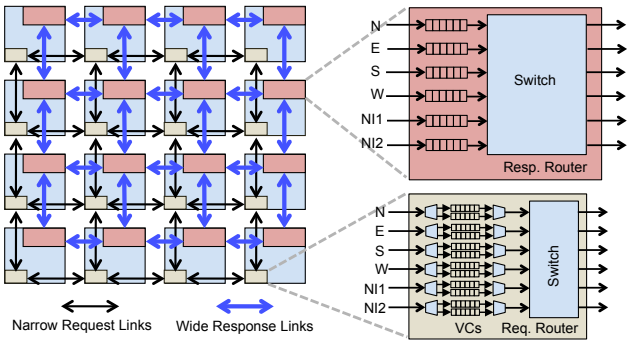

Figure 4: CCNoC with mesh topology. CCNoC uses a pair of asymmetric routers to optimize power and performance for the dominant traffic type of the corresponding network.

of the response traffic. The majority of long responses are messages carrying instruction or read data blocks.

Overall, our results reveal that cache-coherent tiled CMPs exhibit duality in the network traffic when running server workloads, with traffic primarily consisting of short requests and long responses. This distribution differs from that in desktop workloads, where 19% of the request messages are long due to frequent writebacks of dirty cache blocks.

IV. Cache-Coherence Network-on-Chip

In this paper, we propose Cache-Coherence Network-on-Chip (CCNoC), a design that capitalizes on the duality in network traffic of cache-coherent server chips. CCNoC uses two asymmetric networks to achieve higher efficiency as compared to existing designs. Each network is optimized for the dominant traffic type and hence the two networks have different datapath widths, different buffer architectures, and different pipeline lengths. Unlike NoCs for CMPs with simple messaging protocols that favor homogeneous networks, NoCs for cache-coherent CMPs require specialization and heterogeneity to best take advantage of network traffic.

Figure 4 depicts the CCNoC architecture with a mesh topology. Each tile consists of two routers, one of them specialized for the *request* and the other for the *response* network. Because requests primarily consist of short messages, the *request* network can be built with a narrow datapath width to reduce switch area and power with minimal impact on the system performance. The response messages usually carry a cache block and as such benefit from wider channels. The NI connecting the core to the network has physical interfaces to both *request* and *response* routers. The NI routes requests into the narrow *request* network and responses into the wide *response* network. To ensure that the cache coherence protocol is not affected, the receiver NI keeps the order between the request and response messages coming from the same source.

Our system features three types of network messages: (a) normal requests (e.g., read or instruction fetch requests); (b) coherence requests (invalidate or downgrade requests); and (c) responses. Avoiding protocol-level deadlock among different messages classes requires either dedicated virtual channels (VCs) or different physical networks. Thus, single-network topologies require three VCs per input port to guarantee deadlock freedom. The same is true for homogeneous dual-network schemes and heterogeneous organizations that split the traffic based on message size (short, long). In contrast, the proposed CCNoC organization segregates requests and responses via dedicated networks, and as a result, requires virtual channels only on the narrow *request* network to avoid any cyclic dependencies between normal and coherence requests. The wide *response* network is deadlock-free by default, since all response messages are guaranteed to be consumed at the destination. As such, it does not need virtual channels for deadlock freedom, thereby improving area and energy efficiency through reduced buffer requirements.

Specializing the CCNoC networks to traffic class enables further optimizations at the router level. Conventional single- and dual-mesh topologies use a three-stage router pipeline that consists of virtual channel allocation (VA), switch allocation (SA), and switch traversal (ST) stages. In CCNoC, the VC-enabled *request* network features the same router pipeline; however, wormhole routers in the *response* network can be simplified by eliminating the VC allocation stage. This optimization reduces communication delay and diminishes router complexity in the CCNoC *response* network.

While speculation may also be used to reduce router delay [17], existing schemes tend to increase router complexity and adversely impact cycle time. More generally, speculation and other potential microarchitectural mechanisms do not change the benefits that CCNoC provides. The advantages of the CCNoC design hold due to the server traffic characteristics in cache-coherent CMPs.

V. Evaluation

We compare CCNoC to traditional single-network topologies and other dual-network topologies proposed by prior research [1] and show that CCNoC provides significant area and power savings while achieving similar or better performance.

A. Methodology

We compare the performance, area, and energy efficiency of CCNoC to state-of-the-art single- and dual-network topologies. Our reference single-network organization is a mesh with a 176-bit datapath (Mesh-176). We also evaluate a single-network 128-bit mesh (Mesh-128) with the understanding that it offers inferior performance due to lower bisection bandwidth, but better energy efficiency than the wide mesh.

We consider two dual-network schemes. The first is a Homogeneous (i.e., replicated) organization that features two identical 88-bit mesh networks. In this design, traffic is evenly distributed among the two networks to maximize load balance. The other organization is Heterogeneous, featuring a wide network for long messages and a narrow network for short packets. The networks are 112 and 64 bits wide, respectively. Both single- and dual-network NoCs feature a three-stage router pipeline and three VCs per router input port to avoid protocol-level deadlocks.

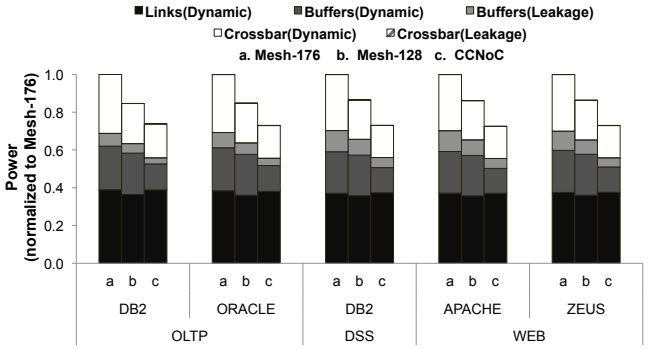

Figure 5: NoC power consumption normalized to Mesh-176 for (a) Mesh-176, (b) Mesh-128, and (c) CCNoC topologies.

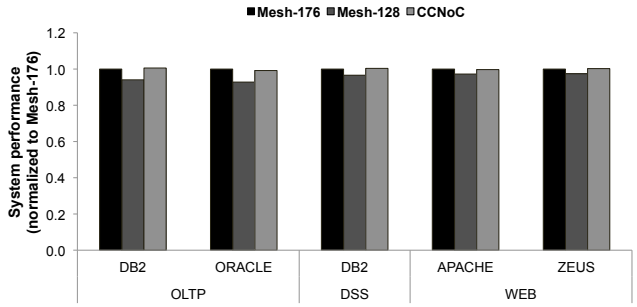

Figure 7: System performance (User-IPC) normalized to Mesh-176.

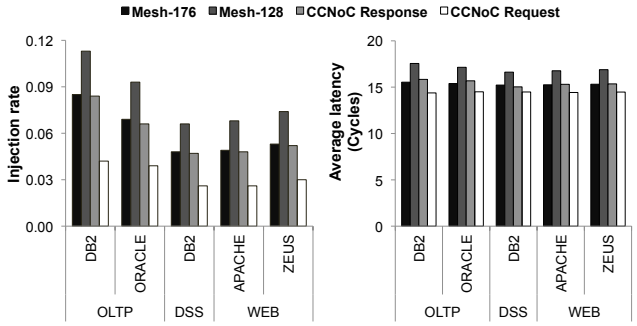

Figure 6: Injection rate (left) and average network latency (right).

Our proposed CCNoC architecture also features a narrow (64-bit) *request* and a wide (112-bit) *response* networks. However, as discussed in Section IV, the networks are specialized. The *request* network carries data request messages and coherence protocol traffic. As such, it requires two VCs per router input port for deadlock avoidance and a three-stage router pipeline similar to reference NoC organizations. The wide *response* network, on the other hand, is dedicated to only one message class. This feature enables a simpler router design based on wormhole flow control with no VCs and a two-stage pipeline.

We use a custom methodology to assess the energy efficiency of the examined NoC organizations. We target 32 nm technology with an on-chip voltage of 0.9 V and a frequency of 2 GHz. We use detailed wire parameters derived from published sources [1, 19] to model the energy expanded in links and router switch fabrics. To reduce link power, we employ differential signaling with 125 mV swing voltage in network channels routed on an intermediate metal layer [18]. Our crossbars are segmented [23] and use full-swing signaling on local wiring with 2x spacing. Each VC uses six flit buffers. We estimate the energy expanded in flit buffers by modifying CACTI 6 [16] to model shallow FIFO configurations representative of typical NoC routers. We also measure leakage power in router buffers and switch fabrics using models derived from CACTI.

In all network organizations, we assume that power gating techniques are applied to eliminate spurious toggling of inactive portions of the datapath. This feature saves power in the cases that the width of the transmitted flit is smaller than the width of the network's datapath [1].

We use Orion 2.0 [9] to estimate the area of router buffers and use our custom methodology to estimate the area of links and crossbar.

B. Comparison to single-mesh NoC designs

We measure the total network power consumption and break it down into major network components: buffers, crossbars, and links. For buffers and crossbars, we track both dynamic and leakage power consumption.

Figure 5 illustrates the network power breakdown for all topologies across our server workloads normalized to Mesh-176. The figure shows that CCNoC reduces total network power by 28% and 18% when compared to Mesh-176 and Mesh-128 respectively. The power savings of CCNoC compared to these network topologies are two-fold.

First, CCNoC requires less total flit storage by virtue of requiring fewer VCs than both Mesh-128 and Mesh-176. This feature reduces combined dynamic and leakage buffer power compared to the reference designs by 42% on average. As buffer power accounts for up to 35% of the network power, the savings are significant.

Second, as noted in Section III-B, a significant fraction of the traffic are short request messages that travel through the narrow *request* network. The compact crossbar in the associated routers diminishes CCNoC's switch power by 40% and 13% when compared to Mesh-176 and Mesh-128, respectively. As crossbars account for 24-30% of the network power in single-mesh topologies, CCNoC considerably reduces their effect on the NoC power consumption.

We evaluate CCNoC's impact on performance by showing both network and system[1] performance across our benchmark suite in Figures 6 and 7, respectively. Figure 6 (left) shows the injection rate (flits/node/cycles) and Figure 6 (right) shows the network latency (i.e., number of cycles to transfer a message from source to destination). Compared to Mesh-176, Mesh-128 has a narrower channel width resulting in a higher effective load and, consequently, higher network latency, resulting in a minor loss of performance of 5%.

[1]We use User-IPC which is proportional to system throughput [24].

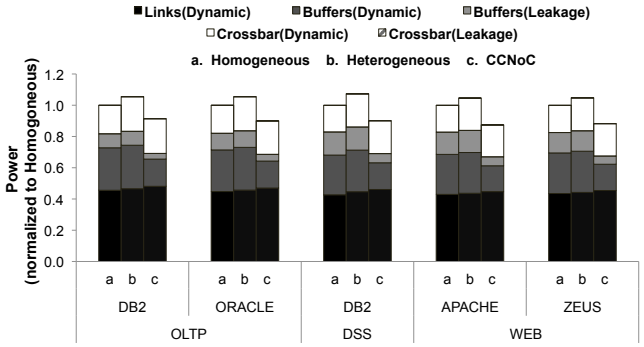

Figure 8: NoC power consumption normalized to Homogeneous for (a) Homogeneous, (b) Heterogeneous, and (c) CCNoC topologies.

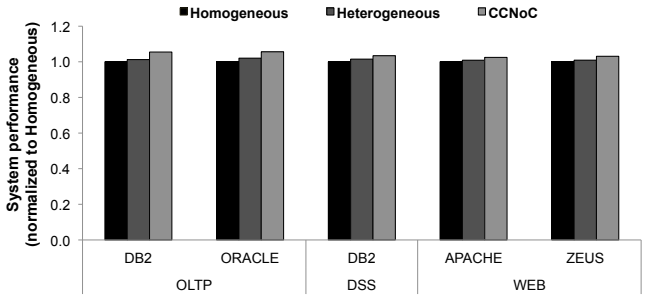

Figure 10: System performance (User-IPC) normalized to Homogeneous.

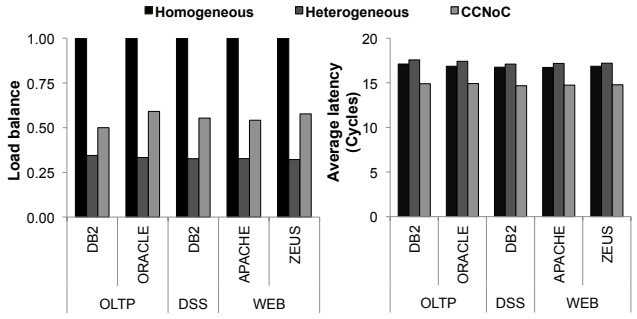

Figure 9: Load balance (left) and average network latency (right).

Long responses in a CCNoC system require one (two) additional flits when compared to Mesh-128 (Mesh-176). However, the extra serialization delay is offset by the reduced load on the *response* network thanks to the separate *request* NoC, as well as the shallower pipeline of wormhole routers in the response network. Figure 6 (right) shows that Mesh-128 and Mesh-176 exhibit, respectively, worse and similar injection rate (and network latency), compared to CCNoC's response network. The request network exhibits lower injection rate, because the majority of injected messages are single-flit. Consequently, the request network latency is slightly lower. Overall, as Figure 7 shows, a CCNoC-enabled CMP matches the system performance of a Mesh-176 organization and outperforms a design based on Mesh-128 by 5%.

C. Comparison to dual-mesh NoC designs

We compare CCNoC to dual-mesh network topologies proposed by Balfour and Dally [1]. As explained in Section V-A, the Homogeneous organization splits the traffic across two identical networks to maximize the load balance and thus reduce network congestion. The Heterogeneous design uses a narrow network to transport short messages and a wide network to transport long messages.

Figure 8 shows that CCNoC consumes 11% and 18% less power than Homogeneous and Heterogeneous organizations, respectively, across our server workloads. The gains are largely due to the efficient buffer architecture of CCNoC. Compared to other dual-network designs, CCNoC features lower VC and buffer requirements across the two networks reducing dynamic and leakage power draw by 44%, on average. Compared to the single-network Mesh-176 design, all three dual-network organizations are effective in reducing switch power by 40-47% (not shown in the figure).

Figure 9 shows the impact of CCNoC on network performance. The figure on the left illustrates the load balance between the two networks. We define load balance as the ratio of the injection rate, in flits, in the narrow network over the injection rate in the wide network. The closer to one this ratio is, the better is the achieved load balance. The figure on the right illustrates the average latency across both networks.

By design, the Homogeneous organization evenly splits the traffic across the two networks and achieves a perfect load balance. In comparison, CCNoC and Heterogeneous designs achieve a load balance ratio of 0.55 and 0.33, respectively, across the two networks. CCNoC significantly improves the load balance over the Heterogeneous design due to the presence of long request (dirty block eviction) messages, which comprise 6% of the request traffic on average. In a CCNoC system, these multi-flit messages travel on the narrow *request* network, which improves its utilization. As dirty block evictions are often not on the critical path, the impact on the system performance is negligible. In contrast, long requests must traverse the wide network in a Heterogeneous design, which diminishes load balance and consumes more power.

In terms of latency, CCNoC bests other dual-network designs due to its efficient architecture. The majority of long messages traverse the wide *response* CCNoC, which improves performance compared to the narrower networks in the Homogeneous design. The performance is also improved against other dual-network organizations thanks to the shallower 2-cycle router pipeline in the CCNoC *response* network.

Figure 10 plots the system performance of the dual-mesh topologies normalized to Homogeneous. CCNoC slightly outperforms both Homogeneous and Heterogeneous NoC designs by 4% and 3%, respectively, thanks to its architecture that specializes the networks to traffic type.

D. Area analysis

Figure 11 plots the NoC area for various single- and dual-network designs. Compared to designs with same bisection bandwidth, CCNoC reduces network area by 31-39%. As buffers occupy 52-58% of the area in single- and dual-network

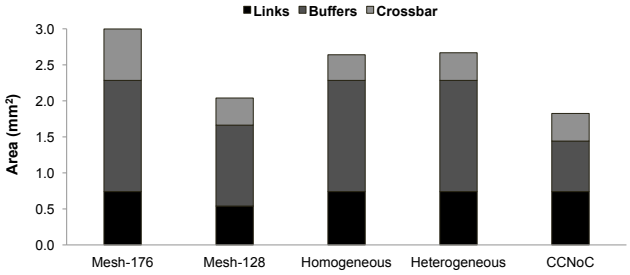

Figure 11: NoC area for various network topologies.

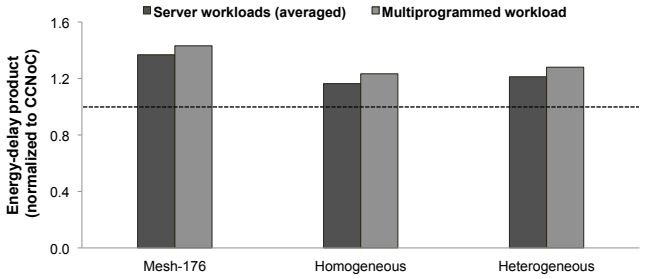

Figure 12: Energy-delay product for server workloads (averaged) and the multiprogrammed workload for various networks.

designs, the gains are largely due to the efficient buffer architecture of CCNoC. In particular, CCNoC reduces the buffer area by 55% compared to Mesh-176, Homogeneous, and Heterogeneous designs. Compared to the single-network Mesh-176 design, all three dual-network organizations are effective in reducing crossbar area by 48-50%.

Compared to a single-network NoC design with smaller bisection bandwidth (i.e., Mesh-128), CCNoC is able to reduce the NoC area by 10%. The gains are largely due to the efficient buffer architecture of CCNoC which offsets the increase in the link area. CCNoC reduces the buffer area by 38%. As crossbar area is proportional to the square of the datapath width, the cost of adding a narrow network is relatively low; as such, CCNoC and Mesh-128 feature similar crossbar footprints.

E. Summary

We summarize our results by calculating the energy-delay product of various network topologies which have same bisection bandwidth. We calculate the energy-delay product of CCNoC, Mesh-176, Homogeneous, and Heterogeneous designs as the product of the total network energy and the CPI (i.e., the inverse of IPC). Figure 12 illustrates the energy-delay product of all network organizations normalized to CCNoC (i.e., dashed line). Across the server workloads, Mesh-176, Homogeneous, and Heterogeneous achieve 37%, 16%, and 21% higher energy-delay product compared to CCNoC.

In workloads with lower network utilization, such as the multi-programmed workload, leakage power constitutes a much higher fraction of the overall buffer power. Because the combined storage footprint of CCNoC networks is lower than that of conventional organizations, CCNoC offers a significant reduction in buffer power. Thus, CCNoC achieves higher network power savings when network utilization is lower. The figure shows that CCNoC improves the energy-delay product by 43%, 23%, and 28% compared to Mesh-176, Homogeneous, and Heterogeneous.

Our findings show that an asymmetric dual-network topology which splits the network traffic according to the coherence protocol patterns is superior to other single- or dual-network topologies. In particular, across our benchmark suite, CCNoC is more energy (area) efficient compared to single- and dual-network topologies by 15-28% (31-39%).

VI. Related Work

The MIT RAW architecture [22] uses four symmetric NoCs (two static networks and two dynamic) which use packet-switched flow control. The Tilera chip [20] extends the MIT RAW architecture and uses five identical wormhole-routed networks to isolate: (a) communication to different sub-systems, (b) memory traffic, and (c) user-specified traffic. In contrast, CCNoC requires only a pair of networks to divide packets at the protocol level. In addition, the networks in CCNoC are asymmetric, yielding greater efficiency and performance through specialization.

Balfour and Dally [1] propose splitting network traffic into two heterogeneous or homogeneous networks to improve performance and power efficiency. The former splits traffic based on message size and the latter strives for load balance across the two networks. Using simple read/write messaging protocols, the authors conclude that the homogeneous design is preferred to a heterogeneous. In this paper, we show that a heterogeneous design which splits network traffic based on message class (request, response) leads to better performance and power efficiency than a homogeneous design for the case of cache-coherent CMPs, as it requires fewer VCs for deadlock avoidance. In addition, we show that a design which splits network traffic based on message class rather than message size leads to better performance and power efficiency, as it achieves better load balance and requires fewer VCs.

Yoon et al. propose using four networks - one for each message class of the MOESI protocol - as an alternative to using virtual channels [25]. In contrast, the dual-network hybrid organization, proposed in our work, makes better use of wire resources by requiring fewer networks, improves network load balance, and boosts performance under a fixed wire budget by supporting a wider response network as compared to a design with multiple dedicated NoCs.

Manevich et al. propose using a hybrid NoC architecture with a bus to broadcast the short commands of the coherence protocol [12]. This work can be considered similar to the heterogeneous approach of Balfour and Dally [1], except that it relies on a bus for the short messages instead of a second network. While appealing for CMPs integrating a small number of cores, a bus-based architecture is hard to scale to many-core configurations that are likely in future server chips.

Prior work has proposed multi-NoC interconnects where one NoC is packet-switched (used for non-localized traffic) and the other is circuit-switched (used for localized traffic) [14]. This optimization targets applications with localized

traffic patterns and is not applicable to server workloads running on cache-coherent CMPs, as the network traffic is uniformly distributed.

Higher radix topologies [4] are considered a viable approach for reducing NoC power consumption. These topologies rely on rich physical connectivity to eliminate a fraction of router traversals. However, the duality-based concept is orthogonal to the network topology and hence the CCNoC concept can be extended to higher radix topologies.

Much research has focused on reducing router buffer power which accounts for a substantial fraction of overall NoC power. The techniques range from those that target a more efficient implementation of buffers [13], bypass buffers [23], or eliminate buffers all together [15]. While many of these techniques increase complexity, they are equally applicable and complementary to CCNoC, as CCNoC targets both buffer and crossbar power consumption.

VII. Conclusion

Server chips are increasingly relying on large number of low-complexity cores to achieve power and performance scalability. In these manycore designs, communication power is becoming a significant fraction of total chip power, calling for improvement in NoC efficiency.

In this work, we introduced *Cache-Coherence Network-on-Chip (CCNoC)*, a heterogeneous dual-network design that capitalizes on the duality in on-chip network traffic observed in cache-coherent multiprocessors and optimizes the routing nodes accordingly. CCNoC employs two asymmetric networks with different datapath widths and router microarchitectures to separate requests and responses. The specialization allows significant power savings with no impact on performance. Through full-system simulation, we showed that CCNoC is more energy efficient than single- and dual-network topologies by 28% and 15%, respectively.

Acknowledgements

The authors would like to thank the members of PARSA at EPFL and the anonymous reviewers for their feedback on drafts of this paper. This work was supported by Eurocloud, Project No 247779 of the European Commission 7th RTD Framework Programme Information and Communication Technologies: Computing Systems, the Cyprus Research Promotion Foundations Grant ΤΠΕ/ΠΛΗΡΟ/0609(BIE)/09 (co-funded by the Republic of Cyprus and the European Regional Development Fund), and the project AdG-246810-NANOSYS of the European Research Council.

References

[1] J. Balfour and W. J. Dally. "Design Tradeoffs for Tiled CMP On-Chip Networks". In *Intl. Conf. on Supercomputing*, June 2006.

[2] G. Blake, R. G. Dreslinski, T. N. Mudge, and K. Flautner. "Evolution of thread-level parallelism in desktop applications". In *Intl. Symp. on Computer Architecture*, June 2010.

[3] M. Ferdman, A. Adileh, O. Kocberber, S. Volos, M. Alisafaee, D. Jevdjic, C. Kaynak, A. D. Popescu, A. Ailamaki, and B. Falsafi. "Clearing the Clouds: A Study of Emerging Scale-Out Workloads on Modern Hardware". In *Intl. Conf. on Architectural Support for Programming Languages and Operating Systems*, March 2012.

[4] B. Grot, J. Hestness, S. W. Keckler and O. Mutlu. "Express Cube Topologies for On-Chip Interconnects". In *Intl. Symp. on High Performance Computer Architecture*, February 2009.

[5] B. Grot, J. Hestness, S. W. Keckler and O. Mutlu. "Kilo-NOC: A Heterogeneous Network-on-Chip Architecture for Scalability and Service Guarantees". In *Intl. Symp. on Computer Architecture*, June 2011.

[6] N. Hardavellas, M. Ferdman, B. Falsafi, and A. Ailamaki. "Reactive NUCA: Near-Optimal Block Placement and Replication in Distributed Caches". In *Intl. Symp. on Computer Architecture*, June 2009.

[7] N. Hardavellas, M. Ferdman, B. Falsafi, and A. Ailamaki. "Toward Dark Silicon in Servers". In *IEEE Micro*, 31(4):615, 2011.

[8] N. Hardavellas, I. Pandis, R. Johnson, N. Mancheril, A. Ailamaki, and B. Falsafi. "Database servers on chip multiprocessors: limitations and opportunities". In *Conference on Innovative Data Systems Research*, January 2007.

[9] A. B. Kahng, B. Li, L.-S. Peh, and K. Samadi. "ORION 2.0: A Fast and Accurate NoC Power and Area Model for Early- Stage Design Space Exploration. In *Design, Automation, and Test in Europe*, April 2009.

[10] P. Lotfi-Kamran, M. Ferdman, D. Crisan, and B. Falsafi. "TurboTag: Lookup Filtering to Reduce Coherence Directory Power". In *Intl. Symp. on Low Power Electronics and Design*, August 2010.

[11] P. Lotfi-Kamran, B. Grot, M. Ferdman, S. Volos, O. Kocberber, J. Picorel, A. Adileh, D. Jevdjic, S. Idgunji, E. Ozer, and B. Falsafi. "Scale-Out Processors". In *Intl. Symp. on Computer Architecture*, June 2012.

[12] R. Manevich, I. Walter, I. Cidon, and A. Kolodny."Best of Both Worlds: A Bus-Enhanced NoC (BENoC)". In *Intl. Symp. on Networks-on-Chip*, May 2009.

[13] G. Michelogiannakis, J. Balfour, and W. J. Dally. "Elastic buffer flow control for on-chip networks". In *Intl. Symp. on High-Performance Computer Architecture*, February 2009.

[14] M. Modarressi, H. Sarbazi-Azad, and M. Arjomand. "An SDM-Based Hybrid Packet-Circuit-Switched On-Chip Network". In *Design, Automation, and Test in Europe Conference*, April 2009.

[15] T. Moscibroda and O. Mutlu. "A case for bufferless routing in on-chip networks". In *Intl. Symp. on Computer Architecture*, June 2009.

[16] N. Muralimanohar, R. Balasubramonian, and N. Jouppi. "Optimizing NUCA Organizations and Wiring Alternatives for Large Caches with CACTI 6.0". In *Intl. Symp. on Microarchitecture*, December 2007.

[17] L.-S. Peh. "Flow Control and Micro-Architectural Mechanisms for Extending the Performance of Interconnection Networks". *PhD Thesis, Stanford University*, August 2001.

[18] D. Schinkel, E. Mensink, E. Klumperink, E. van Tuijl, and B. Nauta. "Low-Power, High-Speed Transceivers for Network-on-Chip Communication". In *IEEE Transactions on VLSI Systems*, 17(1):12-21, 2009.

[19] The International Technology Roadmap for Semiconductors (ITRS). http://www.itrs.net/.

[20] Tilera TILE-Gx 100 core processor. http://www.tilera.com/products/processors/TILE-Gx_Family/.

[21] Transaction Processing Performance Council. http://www.tpc.org/.

[22] M. B. Taylor, J. Kim, J. Miller, D. Wentzlaff, F. Ghodrat, B. Greenwald, H. Hoffmann, P. Johnson, J.-W. Lee, W. Lee, A. Ma, A. Saraf, M. Seneski, N. Shnidman, V. Strumpen, Matt Frank, S. Amarasinghe, and A. Agarwal. "The Raw Microprocessor: A Computational Fabric for Software Circuits and General Purpose Programs". In *IEEE Micro*, 22(2): 25-35, 2002.

[23] H. Wang, L.-S. Peh, and S. Malik. "Power-driven design of router microarchitectures in on-chip networks". In *Intl. Symp. on Computer Architecture*, June 2003.

[24] T. F. Wenisch, R. E. Wunderlich, M. Ferdman, A. Ailamaki, B. Falsafi, and J. C. Hoe. "SimFlex: statistical sampling of computer system simulation". In *IEEE Micro*, 26(4): 18-31, 2006.

[25] Y. J. Yoon, N. Concer, M. Petracca, and L. Carloni. "Virtual channels vs. multiple physical networks: a comparative analysis". In *Design Automation Conference*, June 2010.

[26] J. Zebchuk, V. Srinivasan, M. K. Qureshi, and A. Moshovos. "A Tagless Coherence Directory". In *Intl. Symp. on Microarchitecture*, December 2009.